Funciones

2) Diseñar una función que calcule la potencia enésima de un número, es decir que calcule X n para X, real y n entero

```
//-----------------------------------------------------------------------
float Potencia(float x,int n)
{
    float resultado=1;
    int aux=n;
    if(n<0)
        aux*=-1;
    while(aux)
    {
        resultado*=x;
        aux--;
    }
    if(n<0)
        resultado=1/resultado;
    return resultado;
}
//-----------------------------------------------------------------------
```

4) Utilizando como primitivas las operaciones *, +, - y /, escriba algoritmos que implementen las siguientes primitivas:

- EsDivisiblePor: recibe dos números enteros N y D, y devuelve verdadero si N es divisible por D, y falso en caso contrario.
- ValorAbsoluto: recibe un número entero N, y devuelve el valor absoluto de N.
- RaizCuadradaEntera: recibe un número natural N, y devuelve la raíz entera (sin decimales) de N. Ej: si N es 10, se devuelve 3.
- EsCuadradoPerfecto: recibe un número natural N, y devuelve verdadero si N es cuadrado perfecto; falso en caso contrario.
- Resto: recibe dos nros enteros N y D, y devuelve el resto de dividir N por D.
- Potencia: recibe una base real B y un exponente entero E, y devuelve B elevado a la E.
- CantidadDeCifras: recibe un número entero N, y devuelve la cantidad de dígitos de N (ej: si N = 3421, devuelve 4)

Obs.: el operador “/" representa la división. Cuando se lo aplica a dos enteros Z1 y Z2, Z1 / Z2 corresponde a la división entera entre Z1 y Z2. Cuando se los aplica a reales R1 y R2, la división R1 / R2 corresponde a la división real.

```
//-----------------------------------------------------------------------
char EsDivisiblePor(int N,int D)
{
    while(N>D)
        N-=D;
    if(!N)
        return 1;
```

```
    else
        return 0;
}
//----------------------------------------------------------------------
int ValorAbsoluto(int N)
{
    if(N<0)
        N*=-1;
    return N;
}
//----------------------------------------------------------------------
int RaizCuadradaEntera(int Num)
{
    int ini,fin,resul,rant,cuad;
    ini=0;
    fin=Num/2;
    while(ini<fin)
    {
        rant=resul;
        resul=(ini+fin)/2;
        cuad=resul*resul;
        if(cuad<0)
            fin/=2;
        else
        {
            if(cuad==Num)
                return resul;
            if(cuad>Num)
                fin=resul-1;
            else
                ini=resul+1;
        }
    }
    if(cuad>Num)
        return rant;
    else
        return resul;
}
//----------------------------------------------------------------------
int EsCuadradoPerfecto(int N)
{
    int raiz,cuad;
    raiz=Raiz(N);
    cuad=raiz*raiz;
    if(cuad==N)
        return 1;
    else
        return 0;
}
//----------------------------------------------------------------------
int Resto(int N,int D)
{
    while(N>=D)
        N-=D;
    return N;
}
//----------------------------------------------------------------------
float Potencia(float x,int n)
{
    float resultado=1;
```

```
    int aux=n;
    if(n<0)
        aux*=-1;
    while(aux)
    {
        resultado*=x;
        aux--;
    }
    if(n<0)
        resultado=1/resultado;
    return resultado;
}
//----------------------------------------------------------------------
int CantidadDeCifras(int N)
{
    int cifras=0;
    while(N)
    {
        N/=10;
        cifras++;
    }
    return cifras;
}
//----------------------------------------------------------------------
```

6) El Tercer Teorema de Fermat enuncia que no existen números enteros x, y, z, tales que para un número natural n, n > 2, se verifique que xn+yn=zn. Sin embargo, el matemático francés Pierre Fermat (1601-1665) tan sólo enunció el teorema anterior, pero no pudo demostrarlo. . . (según él, porque la hoja que tenía en ese momento no le alcanzaba). Quizás el teorema sea falso. Para asegurar eso habría que encontrar un contraejemplo, esto es, bastaría encontrar tres números x; y; z y un valor n mayor que 2 tales que xn+yn=zn.
Obs.: para n = 2, lo anterior puede satisfacerse; Ejemplo: 32 +42 = 52).
Escriba un algoritmo que determine si el Tercer Teorema de Fermat se cumple para cualquier entero entre 0 y 20, con n = 3 y n = 4. (Es decir, el algoritmo deberá devolver verdadero si no existen valores x; y; z comprendidos entre 0 y 20, tales que para n=3 y n=4, resulte xn+yn=zn).

```
//----------------------------------------------------------------------
#include <stdio.h>
#include <math.h>
//----------------------------------------------------------------------
struct Dato
{
    int x,y,z,pot;
};
//----------------------------------------------------------------------
struct Dato Fermat(void);
//----------------------------------------------------------------------
int main(void)
{
    struct Dato dato;
```

```
    dato=Fermat();
    if(dato.pot==-1)
        printf("Se cumple el Teorema\n");
    else
    {
        printf("Para x=%d y=%d ",dato.x,dato.y);
        printf("z=%d elevado a %d\n",dato.z,dato.pot);
        printf("No se cumple el Teorema\n\n");
    }
    return 0;
}
//----------------------------------------------------------------------
struct Dato Fermat(void)
{
    struct Dato result;
    float tot,zp;
    for(result.pot=3;result.pot<=4;result.pot++)
    {
        for(result.x=1;result.x<21;result.x++)
        {
            for(result.y=1;result.y<21;result.y++)
            {
               tot=pow(result.x,result.pot)+pow(result.y,result.pot);
               zp=pow(tot,1.0/result.pot);
               if(zp==(int)zp)
               {
                   result.z=zp;
                   return result;
               }
            }
        }
    }
    result.x=-1;
    result.y=-1;
    result.z=-1;
    result.pot=-1;
    return result;
}
//----------------------------------------------------------------------
```

8) Escribir un programa que calcule el número de billetes de 10.000, 5.000, 1,000, así como de monedas de 500, 100, 25, 5 y 1 pesetas para desglosar una cantidad, C, de pesetas (menor de 2.147.483.647), de forma que se necesite la menor cantidad de monedas y billetes de cada tipo.

```
//----------------------------------------------------------------------
#include <stdio.h>
//----------------------------------------------------------------------
struct Billetes
{
    int DiezMil,CincoMil,Mil,Quinientos,Cien,Veinticinco,Cinco,Uno;
};
//----------------------------------------------------------------------
struct Billetes CalculaCantidad(int);
//----------------------------------------------------------------------
int main(void)
{
```

```
    struct Billetes Valor;
    int Monto;
    printf("\nIngrese el monto a calcular ");
    scanf("%d",&Monto);
    Valor=CalculaCantidad(Monto);
    printf("\n Billetes de 10.000 -> %d",Valor.DiezMil);
    printf("\n Billetes de  5.000 -> %d",Valor.CincoMil);
    printf("\n Billetes de  1.000 -> %d",Valor.Mil);
    printf("\n Monedas  de    500 -> %d",Valor.Quinientos);
    printf("\n Monedas  de    100 -> %d",Valor.Cien);
    printf("\n Monedas  de     25 -> %d",Valor.Veinticinco);
    printf("\n Monedas  de      5 -> %d",Valor.Cinco);
    printf("\n Monedas  de      1 -> %d\n\n",Valor.Uno);
}
//----------------------------------------------------------------------
struct Billetes CalculaCantidad(int Monto)
{
    struct Billetes Valor={0,0,0,0,0,0,0,0};
    Valor.DiezMil=Monto/10000;
    Monto%=10000;
    Valor.CincoMil=Monto/5000;
    Monto%=5000;
    Valor.Mil=Monto/1000;
    Monto%=1000;
    Valor.Quinientos=Monto/500;
    Monto%=500;
    Valor.Cien=Monto/100;
    Monto%=100;
    Valor.Veinticinco=Monto/25;
    Monto%=25;
    Valor.Cinco=Monto/5;
    Monto%=5;
    Valor.Uno=Monto;
    return Valor;
}
//----------------------------------------------------------------------
```

10) Escribir un programa que calcule la sumatoria:

$$\sum_{i=1}^{S} (-1)^i * \frac{1}{i^2}$$

Donde S es un número entero positivo introducido por teclado.
Solución: El límite de esa expresión cuando S tiende a infinito es: -0.822467.

```
//----------------------------------------------------------------------
#include <stdio.h>
#include <math.h>
//----------------------------------------------------------------------
float Serie(int);
//----------------------------------------------------------------------
int main(void)
{
    int S;
    printf("\nIngrese el limite de la sumatoria ");
    scanf("%d",&S);
```

```
    printf("\n\nLa sumatoria de los %d ",S);
    printf("primeros terminos es:  %f\n",Serie(S));
    return 0;
}
//----------------------------------------------------------------------
float Serie(int S)
{
    float result=0;
    int i;
    for(i=1;i<=S;i++)
        result+=pow(-1,i)/pow(i,2);
    return result;
}
//----------------------------------------------------------------------
```

Funciones Recursivas

15) Escribir una función que calcule el factorial de un número y utilizar ésta en un programa que muestre el siguiente menú.
1. Factorial de un número
2. Cálculo de e
3. Cálculo de e^x
0. Salir
Nota 1: El cálculo de e debe hacerse con la siguiente expresión matemática:

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots$$

Nota 2: ex puede calcularse mediante la fórmula:

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$$

Nota 3: La precisión con la que se obtiene el resultado (e o e x) depende del último valor añadido en la correspondiente serie.

```
//----------------------------------------------------------------------
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
//----------------------------------------------------------------------
char Menu(void);
void CalculoFactorial(void);
void CalculoE(void);
void CalculoEx(void);
long int Factorial(int);
float ValorEx(float,int);
//----------------------------------------------------------------------
int main(void)
{
    char opcion;
    do
    {
        opcion=Menu();
```

```
        switch(opcion)
        {
        case '1':
            CalculoFactorial();
            break;
        case '2':
            CalculoE();
            break;
        case '3':
            CalculoEx();
            break;
        }
    }
    while(opcion!='0');
    return 0;
}
//----------------------------------------------------------------------
char Menu(void)
{
    char opcion;
    system("CLS");
    printf("\t1 - Calculo de Factorial\n\t2 - Calculo de e\n");
    printf("\t3 - Calculo de e^x\n\t0 - Salir\n\t");
    printf("\n\n\tElija unn opcion: ");
    opcion=getche();
    printf("\n\n");
    return opcion;
}
//----------------------------------------------------------------------
void CalculoFactorial(void)
{
    int Num;
    long int Fact;
    system("CLS");
    printf("\n\t\tIngrese un numero: ");
    scanf("%d",&Num);
    Fact=Factorial(Num);
    printf("\n\n\t\tSu factorial es: %ld",Fact);
    printf("\n\n\tPresione una tecla para continuar");
    getch();
}
//----------------------------------------------------------------------
void CalculoE(void)
{
    int Termino;
    float E;
    system("CLS");
    printf("\n\t\tIngrese la cantidad de terminos: ");
    scanf("%d",&Termino);
    E=ValorEx(1.0,Termino);
    printf("\n\n\t\tEl valor de e calculado con %d ",Termino);
    printf("terminos es: %f",E);
    printf("\n\n\tPresione una tecla para continuar");
    getch();
}
//----------------------------------------------------------------------
void CalculoEx(void)
{
    int Termino;
    float X,Ex;
```

```
    system("CLS");
    printf("\n\t\tIngrese el valor de x: ");
    scanf("%f",&X);
    printf("\n\n\t\tIngrese la cantidad de terminos: ");
    scanf("%d",&Termino);
    Ex=ValorEx(X,Termino);
    printf("\n\n\t\tEl valor de e^%f calculado con ",X);
    printf("%d terminos es: %f",Termino,Ex);
    printf("\n\n\tPresione una tecla para continuar");
    getch();
}
//----------------------------------------------------------------------
long int Factorial(int Num)
{
    long int Fact=1;
    if(Num<=1)
        return Fact;
    Fact*=(Factorial(Num-1)*Num);
}
//----------------------------------------------------------------------
float ValorEx(float X,int Termino)
{
    float Ex=1;
    if(!Termino)
        return Ex;
    return (pow(X,Termino)/Factorial(Termino))+ValorEx(X,Termino-1);
}
//----------------------------------------------------------------------
```

Strings

17) Escribir un programa que pida una frase acabada en un punto y cuente las palabras que contiene.

```
//----------------------------------------------------------------------
#include <stdio.h>
#include <conio.h>
//----------------------------------------------------------------------
int Palabras(char *);
//----------------------------------------------------------------------
int main(void)
{
    char Frase[255];
    int Cant=0;
    printf("Ingrese una frase terminada en un punto \n\n");
    do
    {
        Frase[Cant]=getche();
        Cant++;
    }
    while((Frase[Cant-1]!='.')&&(Cant<255));
    if(Frase[Cant-1]!='.')
        Frase[Cant-1]='.';
    printf("\n\nLa cantidad de palabras es %d\n\n",Palabras(Frase));
}
//----------------------------------------------------------------------
int Palabras(char *Frase)
{
```

```
    int Cant=0;
    while(*Frase!='.')
    {
        while(*Frase!=' '&&*Frase!='.')
            Frase++;
        Cant++;
        if(*Frase!='.')
            Frase++;
    }
    return Cant;
}
//----------------------------------------------------------------------
```

23) Escribir un programa que cuente de un texto introducido por teclado:
- N.º de caracteres en blanco
- N.º de dígitos
- N.º de letras
- N.º de líneas
- N.º de otros caracteres

Nota 1: Se deben crear sendas funciones para comprobar si un carácter es numérico o alfanumérico.
Nota 2: La función getchar() permite leer un carácter de teclado.

```
//----------------------------------------------------------------------
#include <stdio.h>
//----------------------------------------------------------------------
struct Valores
{
    int Blanco,Digitos,Letras,Lineas,Otros;
};
//----------------------------------------------------------------------
void IngresaTexto(char *);
void ObtieneValores(char *,struct Valores *);
int DeterminaCaracteres(char);
void ImprimeResultados(struct Valores);
//----------------------------------------------------------------------
int main(void)
{
    char Texto[255]="";
    struct Valores Datos={0,0,0,0,0};
    IngresaTexto(Texto);
    ObtieneValores(Texto,&Datos);
    ImprimeResultados(Datos);
    return 0;
}
//----------------------------------------------------------------------
void IngresaTexto(char *Texto)
{
    int Largo=0;
    printf("\nIngrese un texto\n");
    do
    {
        *Texto=getchar();
        Largo++;
        Texto++;
```

```
    }
    while(((*(Texto-1)!='\n')||(*(Texto-2)!='\n'))&&(Largo<255));
    *(Texto-1)='\0';
}
//----------------------------------------------------------------------
void ObtieneValores(char *Texto,struct Valores* Datos)
{
    int Aux;
    while(*Texto)
    {
        Aux=DeterminaCaracteres(*Texto);
        switch(Aux)
        {
        case 1:
            Datos->Blanco++;
            break;
        case 2:
            Datos->Digitos++;
            break;
        case 3:
            Datos->Letras++;
            break;
        case 4:
            Datos->Lineas++;
            break;
        case 5:
            Datos->Otros++;
            break;
        }
        Texto++;
    }
}
//----------------------------------------------------------------------
int DeterminaCaracteres(char Caracter)
{
    if(Caracter==' ')
        return 1;
    if((Caracter>='0')&&(Caracter<='9'))
        return 2;
    if((Caracter>='A')&&(Caracter<='Z'))
        return 3;
    if((Caracter>='a')&&(Caracter<='z'))
        return 3;
    if((Caracter=='ñ')||(Caracter=='Ñ'))
        return 3;
    if((Caracter=='á')||(Caracter=='Á'))
        return 3;
    if((Caracter=='é')||(Caracter=='É'))
        return 3;
    if((Caracter=='í')||(Caracter=='Í'))
        return 3;
    if((Caracter=='ó')||(Caracter=='Ó'))
        return 3;
    if((Caracter=='ú')||(Caracter=='Ú'))
        return 3;
    if((Caracter=='ü')||(Caracter=='Ü'))
        return 3;
    if(Caracter=='\n')
        return 4;
    return 5;
```

```
}
//------------------------------------------------------------------------
void ImprimeResultados(struct Valores Datos)
{
    printf("\n\n\nN° de caracteres en blanco: %d",Datos.Blanco);
    printf("\n\nN° de dÍgitos             : %d",Datos.Digitos);
    printf("\n\nN° de letras              : %d",Datos.Letras);
    printf("\n\nN° de lÍneas              : %d",Datos.Lineas);
    printf("\n\nN° de otros caracteres    : %d\n\n",Datos.Otros);
}
```

26) Escribir un programa que pida primero un carácter por teclado y que luego pida una cadena. El programa calculará cuántos caracteres tiene la cadena hasta que lea el carácter introducido primero. Se deberá mostrar un mensaje en pantalla con el número de caracteres introducidos hasta llegar al carácter primero.

```
//------------------------------------------------------------------------
#include <stdio.h>
#include <conio.h>
//------------------------------------------------------------------------
struct Valores
{
    int Blanco,Digitos,Letras,Lineas,Otros;
};
//------------------------------------------------------------------------
void IngresaTexto(char *);
int ObtieneValor(char *,char);
//------------------------------------------------------------------------
int main(void)
{
    char Caracter,Texto[255]="";
    int Cantidad;
    printf("\nIngrese un caracter :");
    Caracter=getche();
    printf("\nIngrese un texto\n");
    IngresaTexto(Texto);
    Cantidad=ObtieneValor(Texto,Caracter);
    printf("\n\nla cantidad de caracteres hasta");
    printf(" %c es: %d\n\n",Caracter,Cantidad);
    return 0;
}
//------------------------------------------------------------------------
void IngresaTexto(char *Texto)
{
    int Largo=0;
    do
    {
        *Texto=getchar();
        Largo++;
        Texto++;
    }
    while((*(Texto-1)!='\n')&&(Largo<255));
    *(Texto-1)='\0';
}
//------------------------------------------------------------------------
int ObtieneValor(char *Texto,char Caracter)
```

```
{
    int Cantidad=1;
    while(*Texto&&(*Texto!=Caracter))
    {
        Cantidad++;
        Texto++;
    }
    return Cantidad;
}
//----------------------------------------------------------------------
```

Estructuras

29) Escribir un programa que lea dos números complejos y permita realizar con ellos las siguientes operaciones aritméticas: suma, resta, multiplicación y división
Nota 1: Se debe crear una función de permita leer un número complejo (su parte real y su parte imaginaria).
Nota 2: Se debe crear una función de permita pasar un número complejo en forma parte real y parte imaginaria a módulo y argumento. Se debe crear una función que permita pasar un número complejo en forma módulo y argumento a parte real y parte imaginaria.
Nota 3: La suma y resta de números complejos se obtiene sumando, o restando, las partes reales y las partes complejas.
El producto de dos números complejos se obtiene multiplicando sus módulos y sumando sus argumentos.
El cociente de dos números complejos se obtiene dividiendo sus módulos y restando sus argumentos.

```
//----------------------------------------------------------------------
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <conio.h>
//----------------------------------------------------------------------
struct Comp
{
    float X,Y;
};
//----------------------------------------------------------------------
int Menu(void);
struct Comp Ingresar(char *);
struct Comp Suma(struct Comp,struct Comp);
struct Comp Resta(struct Comp,struct Comp);
struct Comp Producto(struct Comp,struct Comp);
struct Comp Cociente(struct Comp,struct Comp);
struct Comp CartesianoPolar(struct Comp);
struct Comp PolarCartesiano(struct Comp);
void MostrarResultado(struct Comp);
//----------------------------------------------------------------------
int main(void)
{
    struct Comp Operando1,Operando2,Resultado;
    int Opcion;
    do
```

```
    {
        Opcion=Menu();
        if(Opcion!=5)
        {
            Operando1=Ingresar("primer");
            Operando2=Ingresar("segundo");
        }
        switch(Opcion)
        {
        case 1:
            Resultado=Suma(Operando1,Operando2);
            break;
        case 2:
            Resultado=Resta(Operando1,Operando2);
            break;
        case 3:
            Resultado=Producto(Operando1,Operando2);
            break;
        case 4:
            Resultado=Cociente(Operando1,Operando2);
            break;
        }
        if(Opcion!=5)
            MostrarResultado(Resultado);
    }
    while(Opcion!=5);
    return 0;
}
//----------------------------------------------------------------------
int Menu(void)
{
    int Opcion;
    system("CLS");
    printf("\n\n\t1 - Sumar\n\t2 - Restar");
    printf("\n\t3 - Multiplicar\n\t4 - Dividir");
    printf("\n\t5 - Salir\n\nElija una opción ( 1 - 5 ) : ");
    scanf("%d",&Opcion);
    return Opcion;
}
//----------------------------------------------------------------------
struct Comp Ingresar(char *Texto)
{
    struct Comp Operando;
    printf("\n\nIngrese el %s operando\n\nParte real : ",Texto);
    scanf("%f",&Operando.X);
    printf("\nParte imaginaria : ");
    scanf("%f",&Operando.Y);
    return Operando;
}
//----------------------------------------------------------------------
struct Comp Suma(struct Comp PrimOper,struct Comp SegOper)
{
    struct Comp Resultado;
    Resultado.X=PrimOper.X+SegOper.X;
    Resultado.Y=PrimOper.Y+SegOper.Y;
    return Resultado;
}
//----------------------------------------------------------------------
struct Comp Resta(struct Comp PrimOper,struct Comp SegOper)
{
```

```
    struct Comp Resultado;
    Resultado.X=PrimOper.X-SegOper.X;
    Resultado.Y=PrimOper.Y-SegOper.Y;
    return Resultado;
}
//----------------------------------------------------------------------
struct Comp Producto(struct Comp PrimOper,struct Comp SegOper)
{
    struct Comp Resultado;
    PrimOper=CartesianoPolar(PrimOper);
    SegOper=CartesianoPolar(SegOper);
    Resultado.X=PrimOper.X*SegOper.X;
    Resultado.Y=PrimOper.Y+SegOper.Y;
    Resultado=PolarCartesiano(Resultado);
    return Resultado;
}
//----------------------------------------------------------------------
struct Comp Cociente(struct Comp PrimOper,struct Comp SegOper)
{
    struct Comp Resultado;
    PrimOper=CartesianoPolar(PrimOper);
    SegOper=CartesianoPolar(SegOper);
    Resultado.X=PrimOper.X/SegOper.X;
    Resultado.Y=PrimOper.Y-SegOper.Y;
    Resultado=PolarCartesiano(Resultado);
    return Resultado;
}
//----------------------------------------------------------------------
void MostrarResultado(struct Comp Operando)
{
    printf("\n\n\tEl resultado es:\n\n");
    printf("\n\t\tx= %f\t\ty= %f\n\n",Operando.X,Operando.Y);
    printf("\n\nPresione una tecla para continuar");
    getch();
}
//----------------------------------------------------------------------
struct Comp CartesianoPolar(struct Comp Operando)
{
    struct Comp Resultado;
    Resultado.X=sqrt(pow(Operando.X,2)+pow(Operando.Y,2));
    Resultado.Y=atan2(Operando.Y,Operando.X);
    return Resultado;
}
//----------------------------------------------------------------------
struct Comp PolarCartesiano(struct Comp Operando)
{
    struct Comp Resultado;
    Resultado.X=Operando.X*cos(Operando.Y);
    Resultado.Y=Operando.X*sin(Operando.Y);
    return Resultado;
}
//----------------------------------------------------------------------
```

Archivos

34) Leer completamente un fichero de texto, carácter a carácter (o en cantidades mayores, para que sea más rápido). El programa

debe contar las vocales, los caracteres alfabéticos y los dígitos que hay en el fichero.

```
//-----------------------------------------------------------------------
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>
//-----------------------------------------------------------------------
void ProcesarArchivo(char *,int *,int *,int *);
void ImprimirResultados(int,int,int);
void DeterminarTipo(char,int *,int *,int *);
//-----------------------------------------------------------------------
int main(void)
{
    char Nombre[255]="";
    int Vocales=0,Letras=0,Digitos=0;
    do
    {
        system("CLS");
        printf("\n\nIngrese el nombre del archivo : ");
        gets(Nombre);
    }
    while(!strcmp(Nombre,""));
    ProcesarArchivo(Nombre,&Vocales,&Letras,&Digitos);
    ImprimirResultados(Vocales,Letras,Digitos);
    return 0;
}
//-----------------------------------------------------------------------
void ProcesarArchivo(char *Nombre,int *Vocal,int *Letra,int *Digit)
{
    FILE *fp;
    char Caract;
    if(!(fp=fopen(Nombre,"r")))
    {
        printf("\n\nError de apertura del archivo : %s",Nombre);
        printf("\nPresione una tecla para continuar\n");
        getch();
        exit(1);
    }
    while(!feof(fp))
    {
        Caract=fgetc(fp);
        DeterminarTipo(Caract,Vocal,Letra,Digit);
    }
    fclose(fp);
}
//-----------------------------------------------------------------------
void ImprimirResultados(int Vocal,int Letra,int Digit)
{
    printf("\n\nLa cantidad de Vocales es : %d",Vocal);
    printf("\nLa cantidad de Letras es  : %d",Letra);
    printf("\nLa cantidad de Digitos es : %d\n",Digit);
}
//-----------------------------------------------------------------------
void DeterminarTipo(char Caract,int *Vocal,int *Letra,int *Digit)
{
    if(((Caract>='a')&&(Caract<='z'))||((Caract>='A')&&(Caract<='Z')))
    {
```

```
        if((Caract=='a')||(Caract=='A')||(Caract=='e'))
        {
            (*Vocal)++;
            return;
        }
        if((Caract=='E')||(Caract=='i')||(Caract=='I'))
        {
            (*Vocal)++;
            return;
        }
        if((Caract=='o')||(Caract=='O')||(Caract=='u'))
        {
            (*Vocal)++;
            return;
        }
        if(Caract=='U')
        {
            (*Vocal)++;
            return;
        }
        (*Letra)++;
        return;
    }
    if((Caract>='0')&&(Caract<='9'))
    {
        (*Digit)++;
        return;
    }
}
//--------------------------------------------------------------------
```